從攻擊視角認識 Web 弱點 － 以 OWASP Juice Shop 為例

2025年7月30日

◦ 國立中央大學
許時準組長

台灣大學 - 找出OWASP Juice Shop的漏洞練習

課程網址https://reurl.cc/2K53qE

課程講義下載
https://in.ncu.edu.tw/~center20/20250730.pdf

課程目標

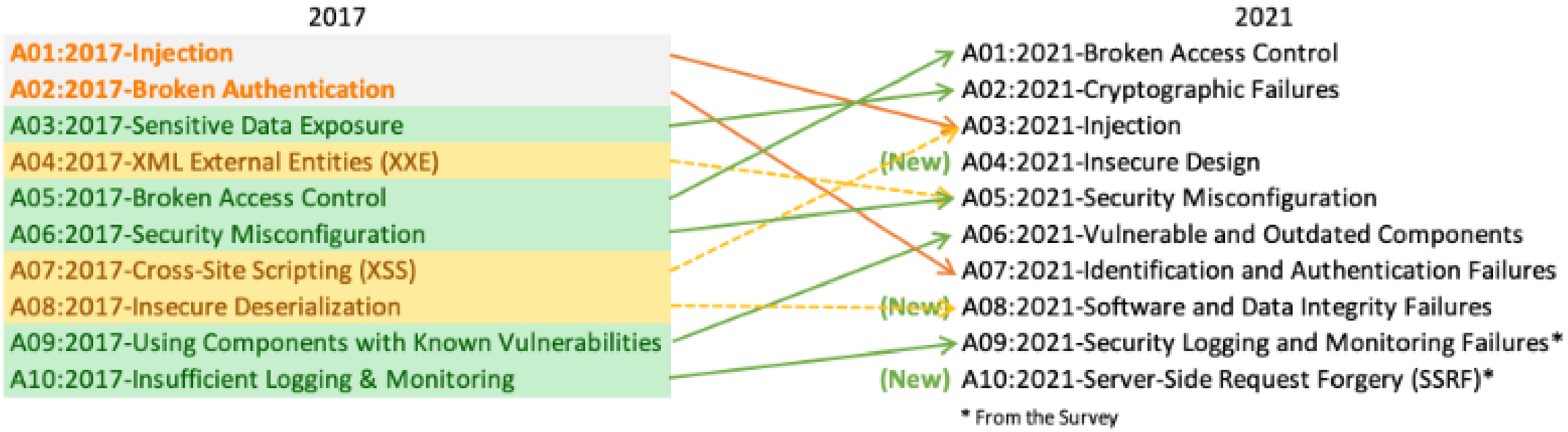
了解常見的Web安全漏洞

實際演練資安攻防

大綱

- OWASP TOP 10
- 網站靶機安裝
- 資安攻防實作
- 結論

第一部分：
OWASP TOP 10



https://owasp.org/Top10/

1:

A01:2021 – 權限控制失效

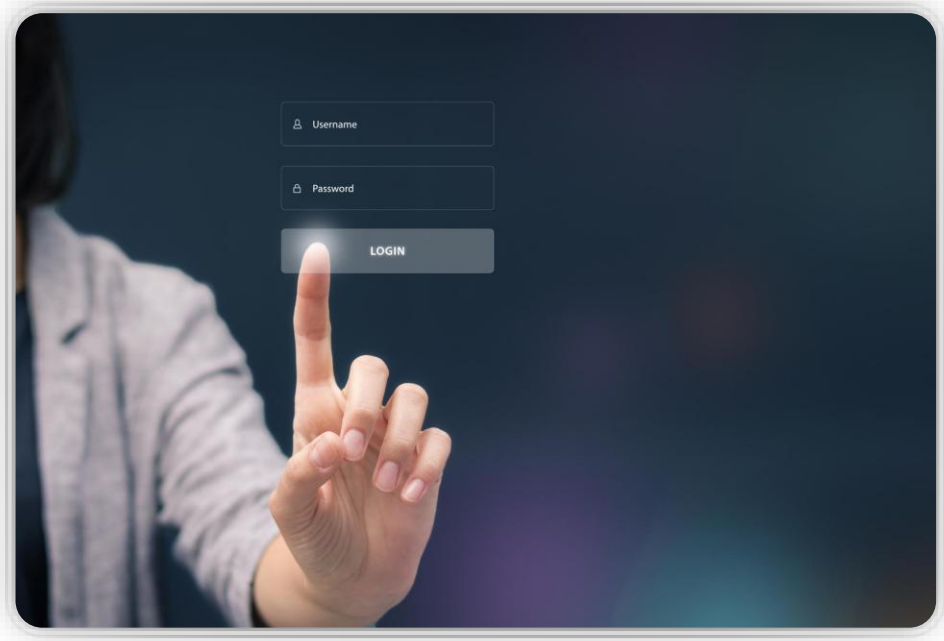
修改URL、內部應用程式狀態或HTML頁面來繞過存取控制檢查。

容許主鍵被更改為其他用戶的記錄，允許查看或編輯其他人的帳戶。

特權提升。未登入即成為用戶，或以用戶身份登入即成為管理員。

https://owasp.org/Top10/

如何預防權限控制失效



1. 存取控制僅在受信任的伺服器端檢查。
2. 除公開的資源外，預設為拒絕存取。
3. 停用Web伺服器目錄列表，並確保檔案中繼資料不在web根目錄中。
4. 記錄存取控制失效，並警示管理員。
5. 對API和控制器存取進行流量限制。

https://owasp.org/Top10/

2:

A02:2021 – 加密機制失效

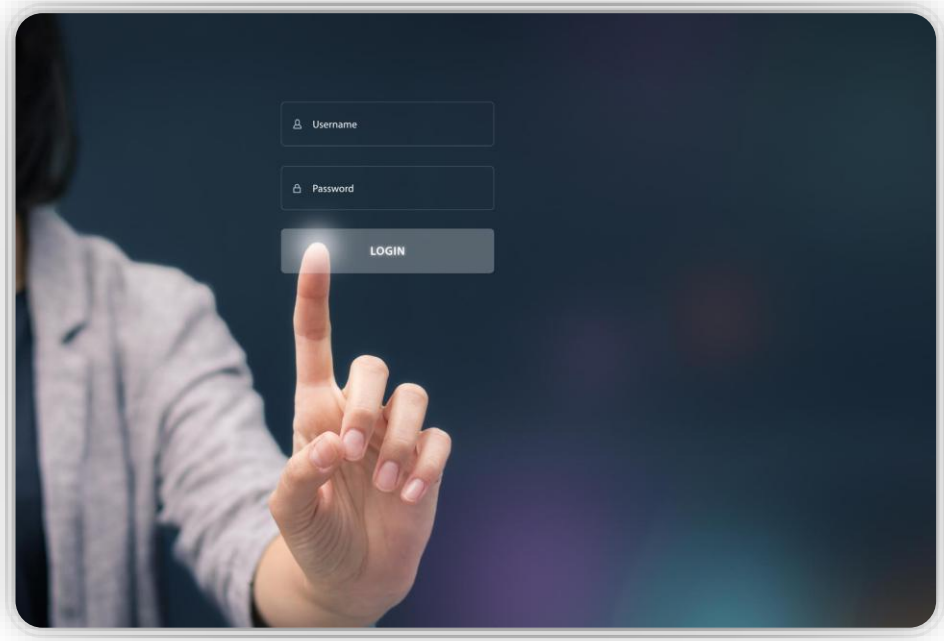
資料是否以明碼傳輸? 像是HTTP, SMTP, FTP等協定。

是否有任何老舊或脆弱的加密演算法被預設使用或存在於較舊的程式碼?

是否有任何預設的加密金鑰被使用、脆弱的加密金鑰被重複使用?

https://owasp.org/Top10/

如何預防加密機制失效



1. 對應用程式處理、儲存、傳輸的資料進行分類，辨識哪些為敏感性資料，依照分類執行對應的控制措施。

2. 確保將所有靜態的敏感性資料加密。

3. 確認使用最新版且標準的強演算法、協定及金鑰。

4. 使用安全的協定加密傳輸中的資料。

https://owasp.org/Top10/

3:

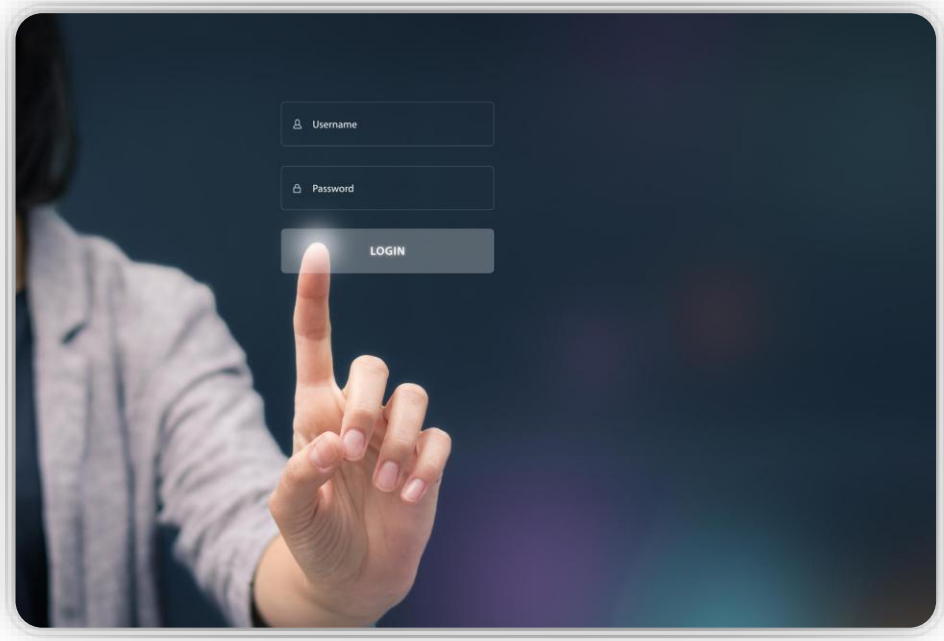
A03:2021 – 注入式攻擊

應用程式未驗證、過濾或清理使用者提供的資料。

在動態查詢、命令或儲存的程序，SQL、指令或儲存的程序中，直接使用或連結惡意資料。

https://owasp.org/Top10/

如何預防注入式攻擊



1. 需要將命令與查詢資料分開，以防止注入式攻擊。

2. 使用正面或白名單在伺服器端驗證輸入的資料。

3. 對於任何剩餘的動態查詢，在轉譯中使用特殊符號進行查詢將對查詢語法帶來不同的涵義。

https://owasp.org/Top10/

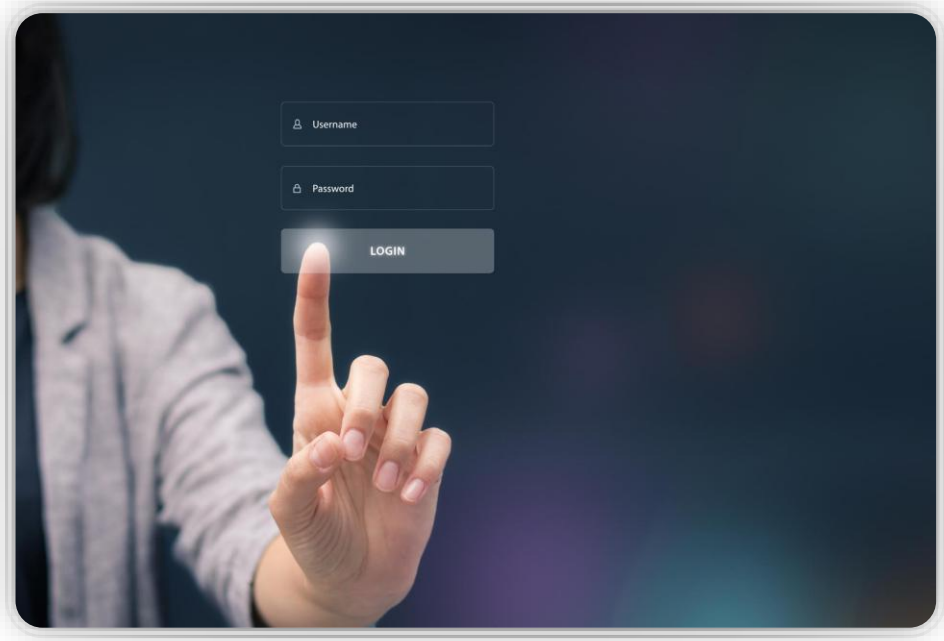
4:

A04:2021 – 不安全設計

- 不安全設計是一個廣泛的類別呈現許多不同的弱點，代表為"缺乏或無效的控制設計"。 缺乏不安全設計是指没有控制措施。

https://owasp.org/Top10/

如何預防不安全設計



1. 建立與使用安全開發生命週期，協同專業人士來評估與設計安全與隱私相關的控制措施。
2. 建立與使用安全設計模式的函式庫或是已完成可使用的元件。
3. 撰寫單元測試與整合測試來驗證所有的關鍵流程。

https://owasp.org/Top10/

5:

A05:2021 – 安全設定缺陷

只啟用必要的功能或是安裝 (例如，不必要的port，服務，頁面，帳號，或是特權)。

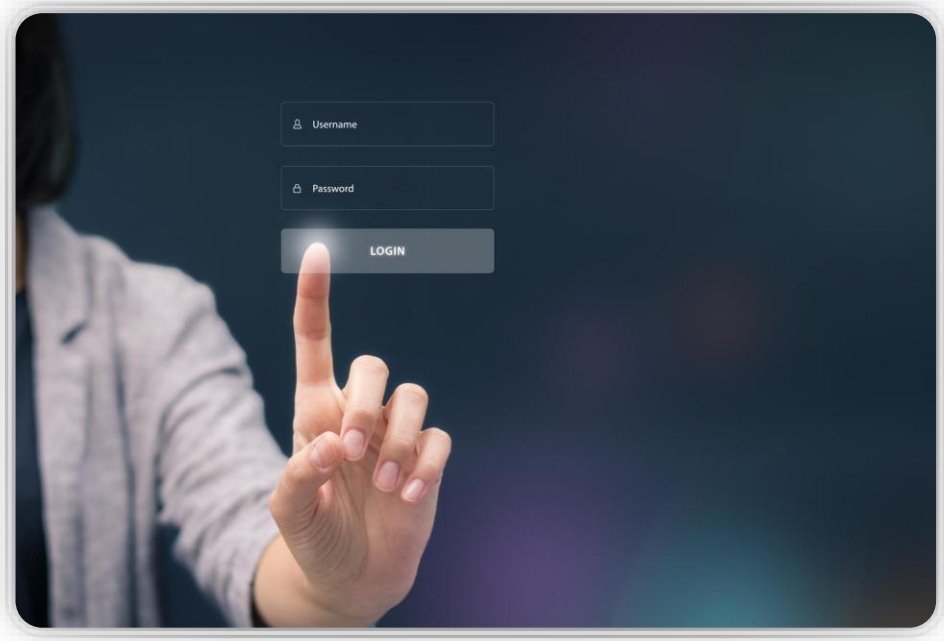
預設帳號與密碼還可使用，並且未更改。

因錯誤處理而暴露出的堆疊追蹤，或是向使用者，暴露出過多的錯誤警告資訊

因為系統升級，導致最新的安全功能被關閉，或是造成不安全的設定

https://owasp.org/Top10/

如何預防安全設定缺陷



1. 確保每個用戶、進程或服務僅獲得其所需的最小權限，以降低攻擊風險。
2. 僅啟用需要的服務和功能，關閉不必要的服務。
3. 使用強大的密碼政策，要求用戶使用長、複雜、並且定期更換的密碼。
4. 實施多因素身份驗證。

https://sites.google.com/gs.ncku.edu.tw/vulnerability-patching-guide/%E5%BC%B1%E9%BB%9E%E4%BF%AE%E8%A3%9C/owasp-top10-web-patch

6:

A06:2021 – 易受攻擊和已淘汰的組件

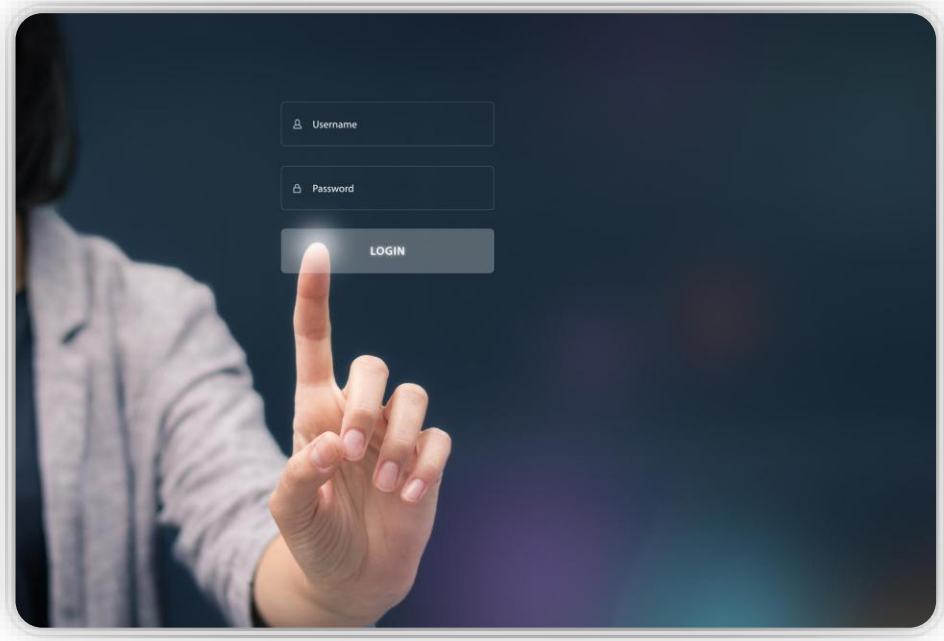
過時的組件的版本（用戶端和伺服器端）。 這包括直接使用的組件以及嵌入的相依套件。

已不支援或已淘汰的作業系統、網頁/應用程式伺服器、資料庫、應用程式、 API 以及所有組件、執行環境和程式庫。

軟體開發人員未測試更新、升級或修補後程式庫的相容性。

https://owasp.org/Top10/

如何預防易受攻擊和已淘汰的組件



1. 刪除未使用的相依套件、不必要的功能、組件、檔案及文件。

2. 持續使用版控工具來盤點客戶端和伺服器端組件（例如框架、程式庫）及相依組件的版本。

3. 僅透過官方提供的安全連結來取得組件。 優先選擇已簽署的更新包，以降低更新包被加入惡意組件的可能。

https://owasp.org/Top10/

7:

A07:2021 – 認證及驗證機制失效

允許像是攻擊者已經擁有有效用戶名稱和密碼列表的撞庫自動化攻擊。

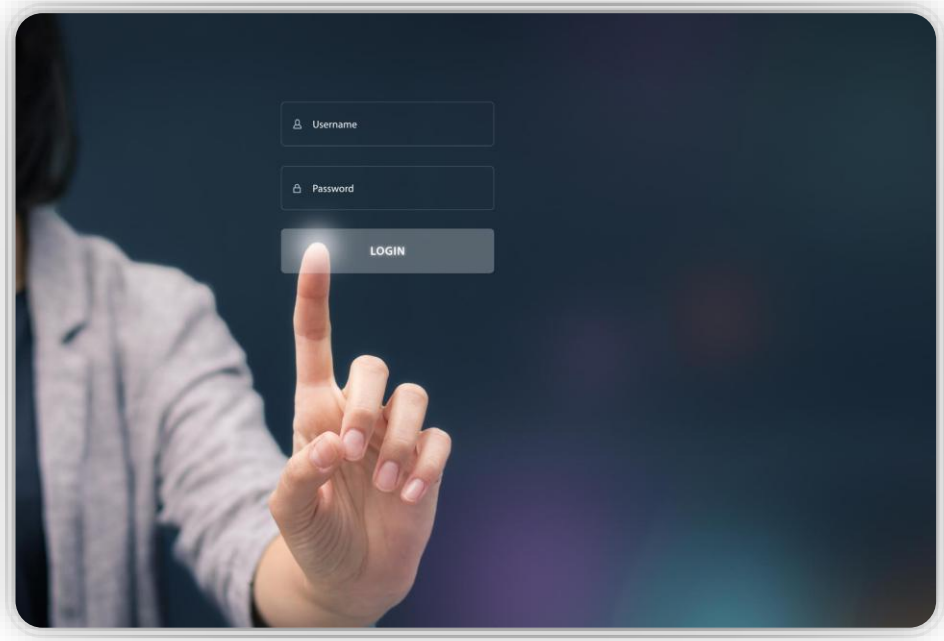
允許預設、脆弱、常見的密碼，像是"Password"或"admin/admin"。

使用脆弱或無效的認證資訊回復或忘記密碼的流程。

將密碼使用明碼、加密或較脆弱雜湊法的方式儲存。

https://owasp.org/Top10/

如何預防認證及驗證機制失效



1. 實作多因子認證來防止自動化撞庫攻擊、暴力破解、以及遭竊認證資訊被重複利用的攻擊。

2. 不要交付或部署任何預設的認證資訊，特別是管理者。

3. 實作脆弱密碼的檢查。

4. 限制或增加失敗登入嘗試的延遲。

https://owasp.org/Top10/

8:

A08:2021 – 軟體及資料完整性失效

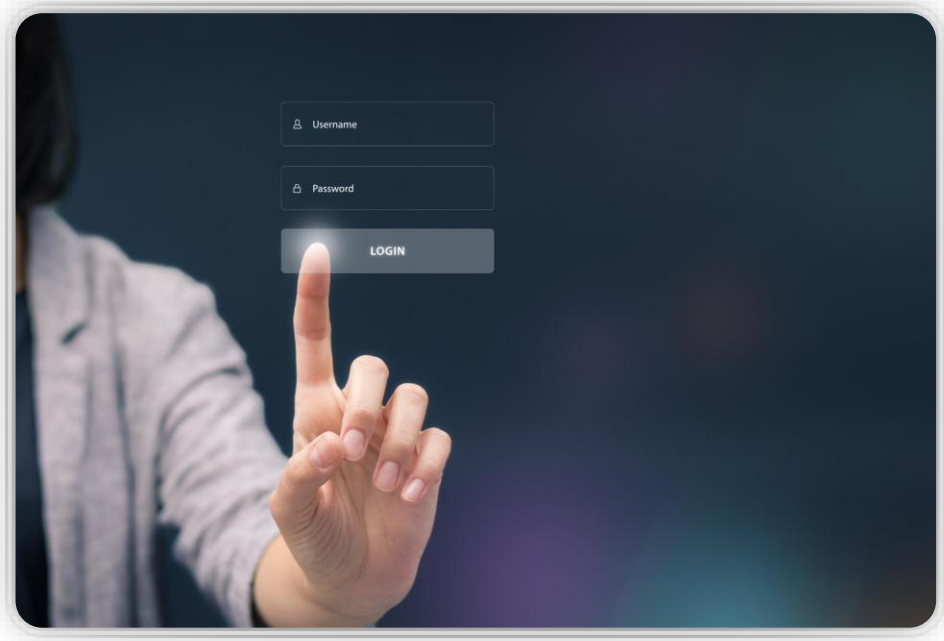
程式碼或基礎架構未能保護軟體及資料之完整性受到破壞。

應用程式依賴來自於不受信任來源，典藏庫及內容遞送網路之外掛，函式庫或模組。

自動更新功能在缺乏充足完整性驗證功能時就下載並安裝更新到處於安全狀態下的應用程式。

https://owasp.org/Top10/

如何預防軟體及資料完整性失效



1. 利用完整性檢查或數位簽章來偵測竄改或重放攻擊。

2. 利用數位簽章或類似機制確保軟體或資料來自預期之提供者

3. 確保函式庫及從屬套件，例如npm或Maven，是從受信任的典藏庫取得。

4. 使用軟體供應鏈安全工具(例如OWASP Dependency Check 或 OWASP CycloneDX)確保元件沒有已知弱點。

https://owasp.org/Top10/

9:

A09:2021 – 資安記錄及監控失效

可稽核事件未記錄，如登入成功，登入失敗及高價值交易。

警告或錯誤發生時未產生，產生不充足或產生不明確日誌。

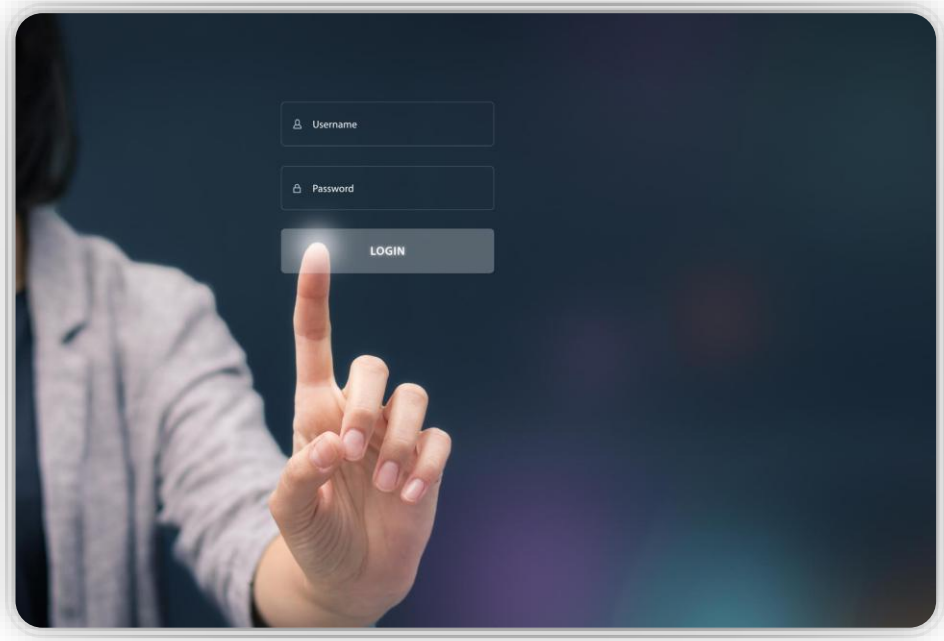
未監控應用程式或應用程式介面(API)日誌中的可疑活動。

日誌僅儲存於本地端。

滲透測試及DAST工具(如OWASP ZAP)掃描沒有觸發告警。

https://owasp.org/Top10/

如何預防資安記錄及監控失效



1. 確保記錄所有登入，存取控制及伺服器端輸入驗證之失敗，日誌應存留充足時間以利未來可能之鑑識分析要求。

2. 確保日誌格式符合一般日誌管理系統常用格式。

3. 確保日誌經正確編碼以防止遭受注入攻擊或日誌/監控系統遭受攻擊。

4. 建立或導入事件應變及復原計畫。

https://owasp.org/Top10/

10:

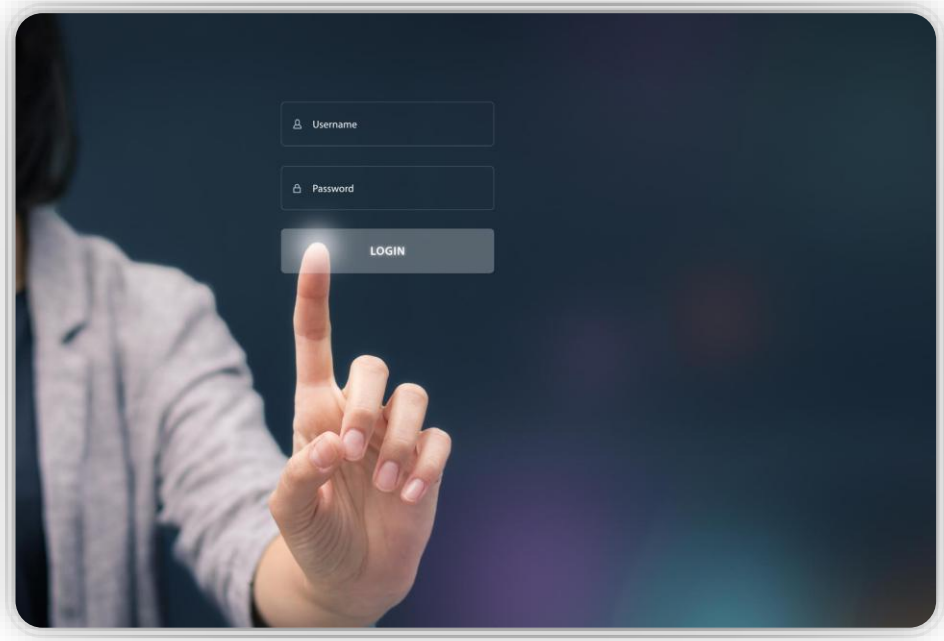
A10:2021 – 伺服端請求偽造

1. 當網頁應用程式正在取得遠端資源，卻未驗證由使用者提供的網址，此時就會發生偽造伺服端請求。

2. 即便有防火牆、VPN或其他網路ACL保護的情況下，攻擊者仍得以強迫網頁應用程式發送一個經過捏造的請求給一個非預期的目的端。

https://owasp.org/Top10/

如何預防伺服端請求偽造



1. 於防火牆政策或於網路存取控制規則實施"預設全拒絕(deny by default)" ，以封鎖全部來自外部之網路流量

2. 過濾並驗證來自於用戶端提供之全部輸入

3. 以正面表列方式列出URL、port、目的地清單

4. 停用HTTP重新導向

https://owasp.org/Top10/

第二部分：

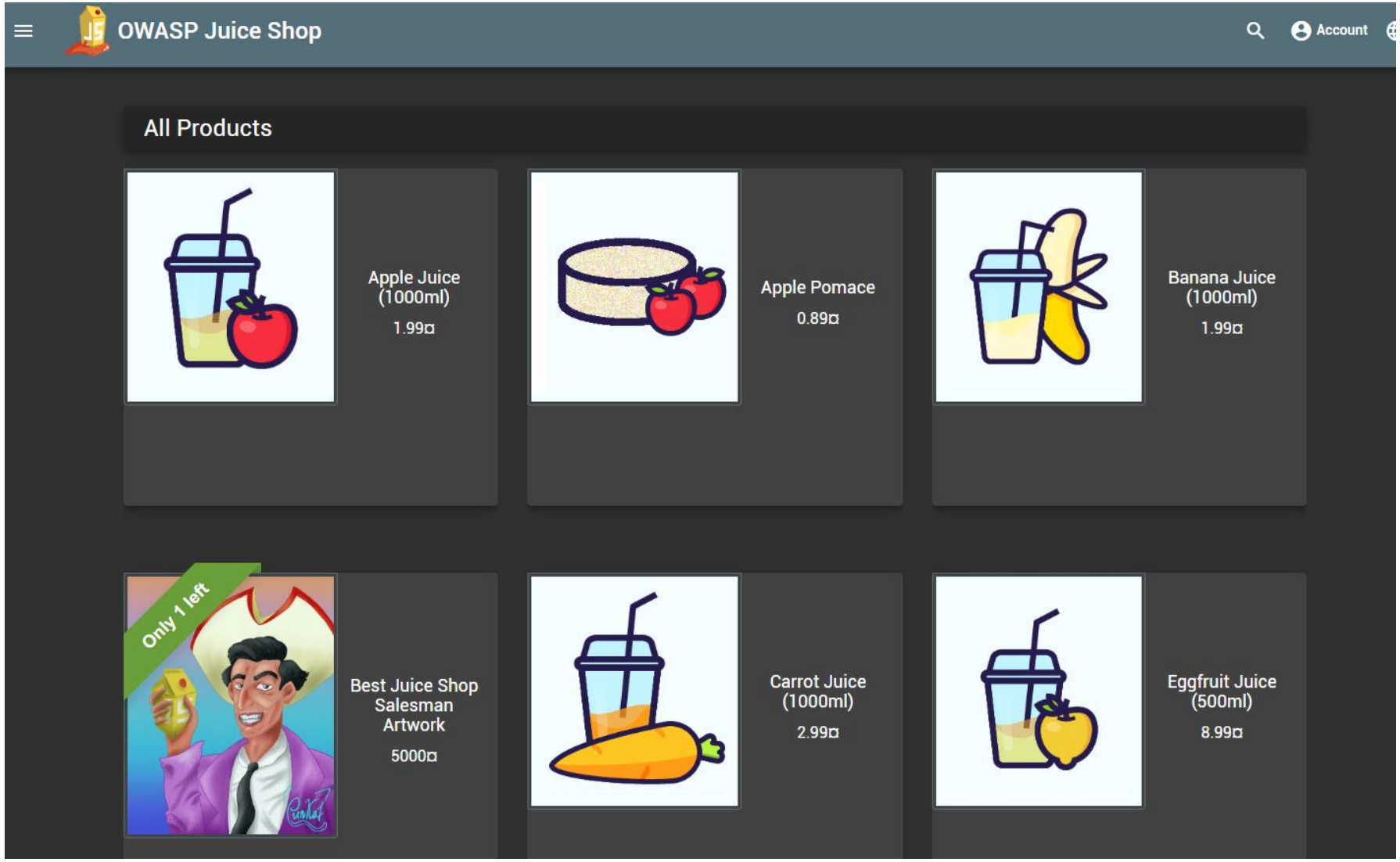
網站靶機安裝

環境設置

安裝與啟動OWASP Juice Shop

確認環境運行正常

http://127.0.0.1:3000/



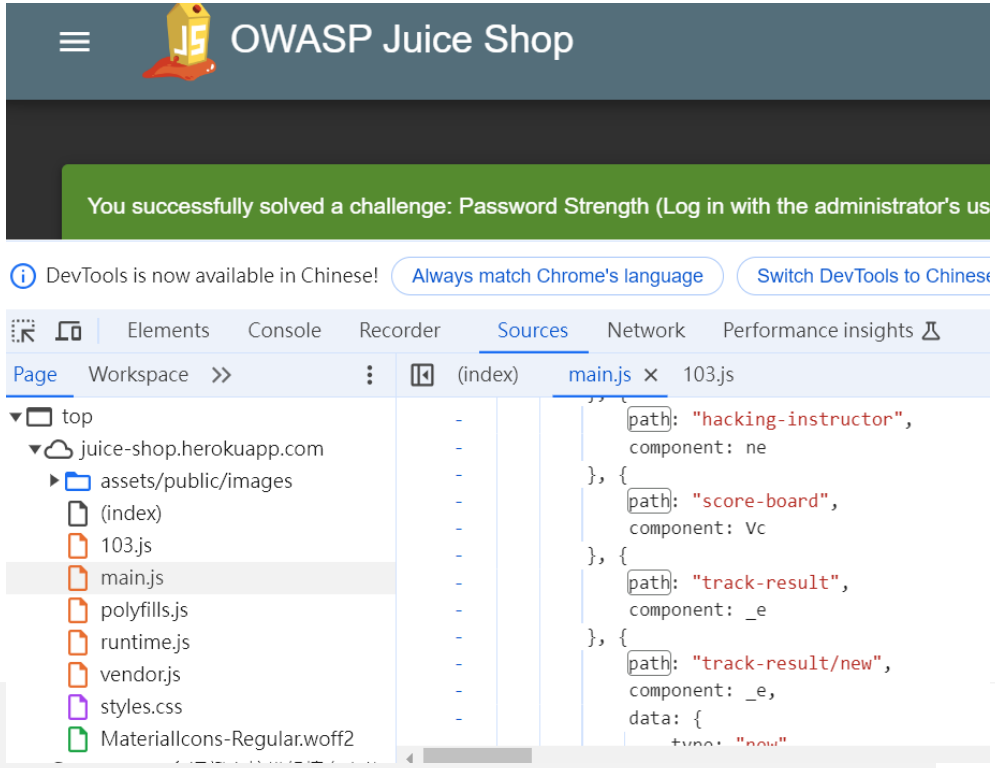
第三部分：

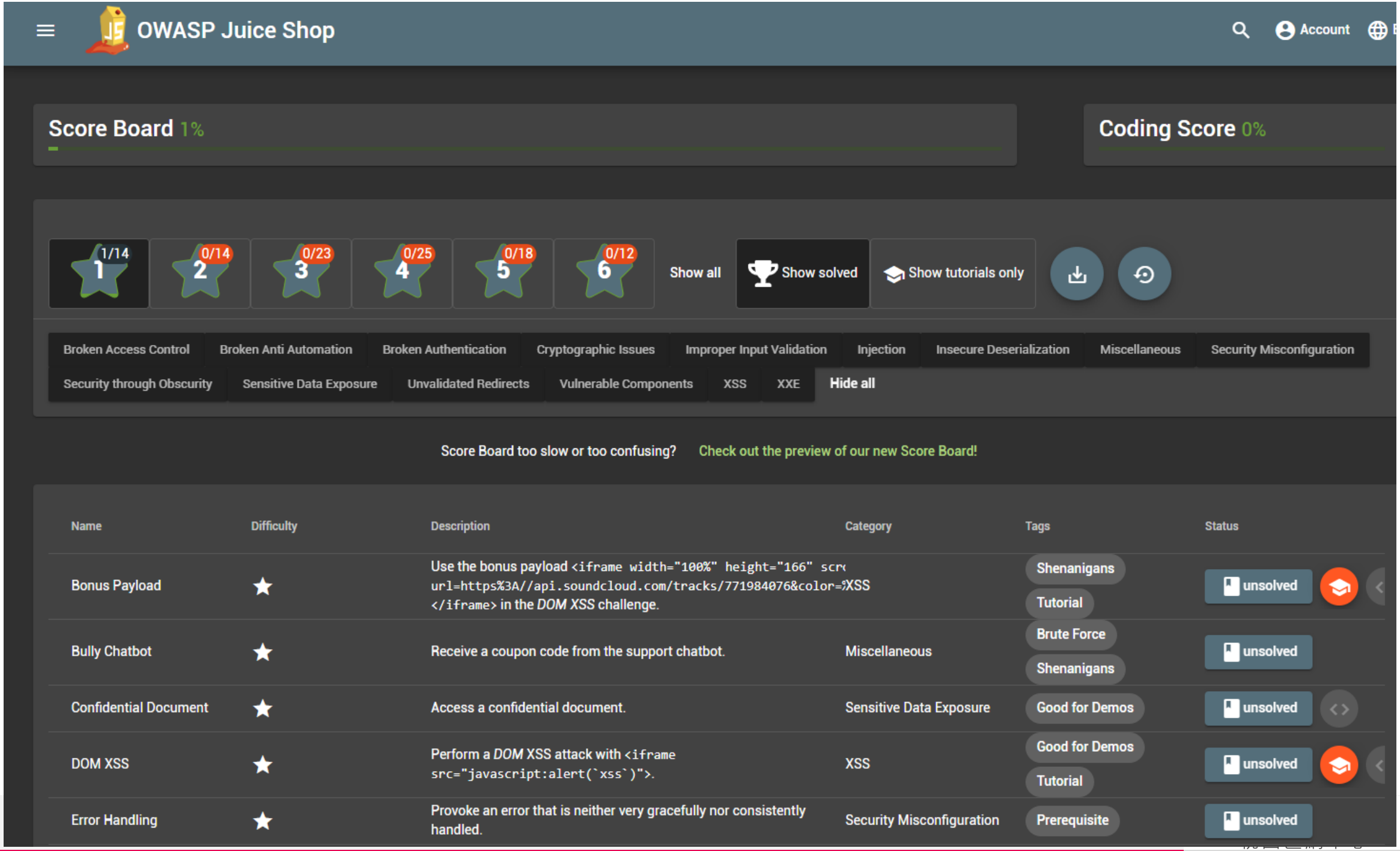
資安攻防實作

- 模擬攻擊
  - 嘗試通過各種方式找尋可能包含敏感資料的URL
  - http://127.0.0.1:3000/

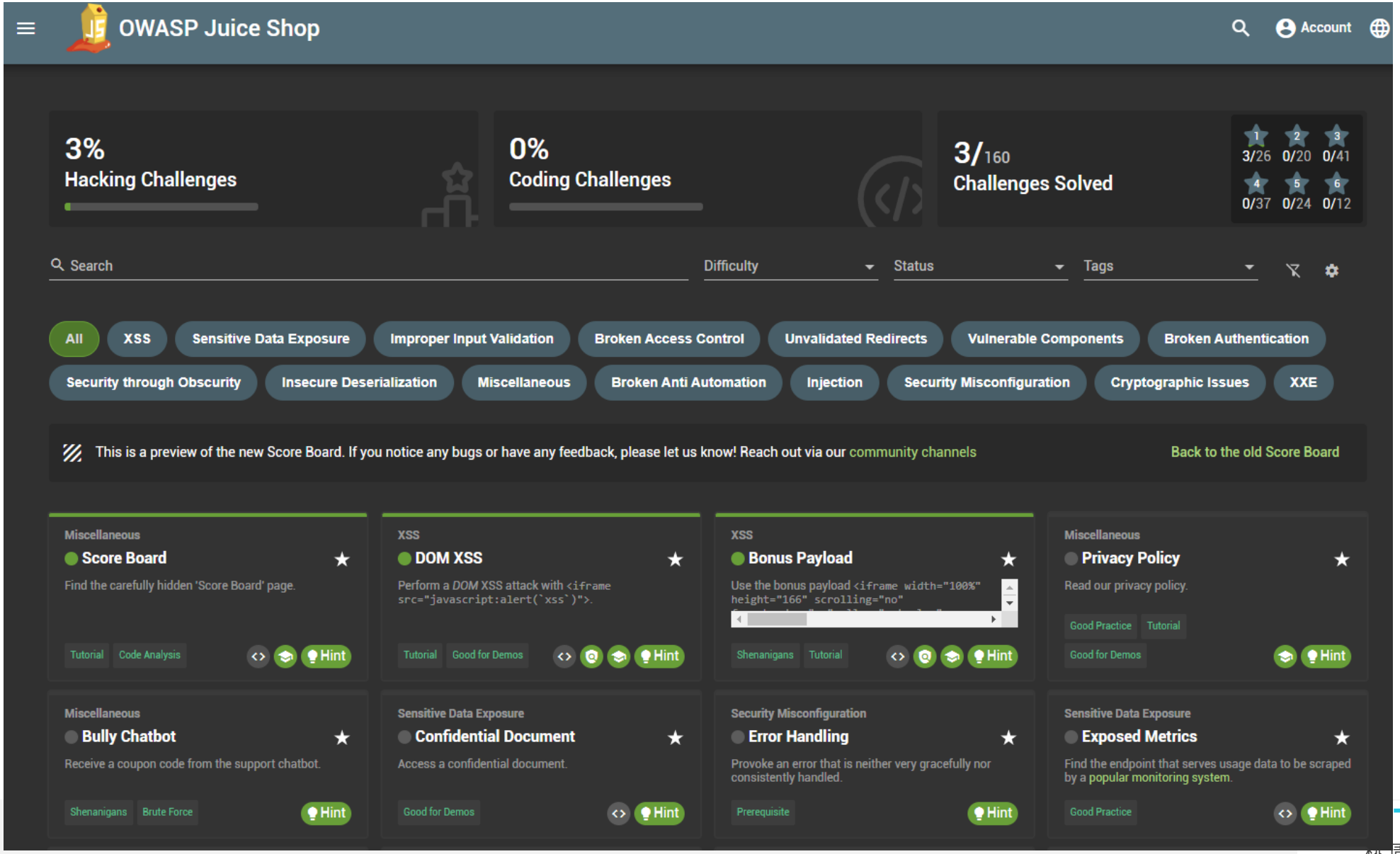
起手式：找到隱藏的記分板

- 開啟F12檢視網站原始碼，包含js檔
- 找尋隱藏其中的URL和線索
- 可利用DIRB 網站目錄掃描等工具




OWASP Juice Shop
Account
Score Board 1%
Coding Score 0%
1/14
0/14
0/23
0/25
0/18
0/12
Show all
Show solved
Show tutorials only
Broken Access Control
Broken Anti Automation
Broken Authentication
Cryptographic Issues
Improper Input Validation
Injection
Insecure Deserialization
Miscellaneous
Security Misconfiguration
Security through Obscurity
Sensitive Data Exposure
Unvalidated Redirects
Vulnerable Components
XSS
XXE
Hide all
Score Board too slow or too confusing?
Check out the preview of our new Score Board!
Name
Difficulty
Description
Category
Tags
Status
Bonus Payload
Use the bonus payload <iframe width="100%" height="166" scr(url=https%3A//api.soundcloud.com/tracks/771984076&color=%XSS </iframe> in the DOM XSS challenge.
Shenanigans
Tutorial
unsolved
Bully Chatbot
Receive a coupon code from the support chatbot.
Miscellaneous
Brute Force
Shenanigans
unsolved
Confidential Document
Access a confidential document.
Sensitive Data Exposure
Good for Demos
unsolved
DOM XSS
Perform a DOM XSS attack with <iframe src="javascript:alert(`xss`)">.
XSS
Good for Demos
Tutorial
unsolved
Error Handling
Provoke an error that is neither very gracefully nor consistently handled.
Security Misconfiguration
Prerequisite
unsolved

切換新介面 http://127.0.0.1:3000/#/score-board-preview



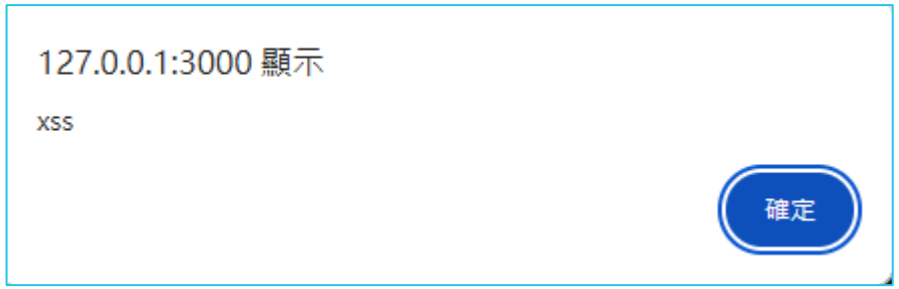
XSS
跨網站指令碼攻擊

- 跨網站指令碼（Cross-site scripting，XSS）攻擊漏洞，允許駭客輸入惡意程式碼
- https://www.ithome.com.tw/news/139205

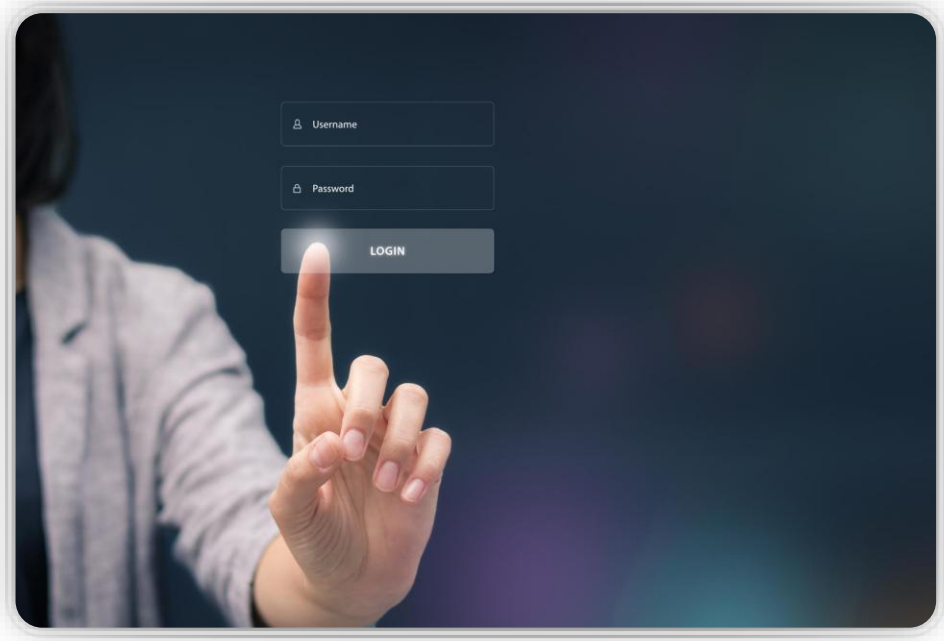
LAB 1:

DOM XSS

- DOM（Document Object Model）型 XSS 是跨站腳本攻擊（Cross-Site Scripting, XSS）的一種，攻擊發生在 用戶端的 JavaScript 環境中。
- 主要是因為開發者在前端程式碼中，不當使用來自 URL、表單、Cookie 等位置的資料，並直接插入頁面或執行程式碼。這類攻擊不經過伺服器端處理，因此 傳統的伺服器端過濾或日誌系統不容易察覺這類攻擊。
- 題目: 如何執行XSS 攻擊?
- Hint : 輸入惡意程式碼
  <iframe src="javascript:alert(`xss`)">.



如何預防DOM XSS攻擊



1. 避免使用 innerHTML、document.write()、eval() 等危險方法。
2. 使用文字插入 API：如 textContent 或 setAttribute，不會解析 HTML。
3. 對輸入資料進行適當的 HTML 轉義（escaping）或清理（sanitization）。
4. 使用現代框架（如 React、Vue）來處理 DOM，這些框架自動對資料做 escape 處理。
5. 導入 CSP（Content Security Policy），防止不明的 JavaScript 執行。

LAB 2 :

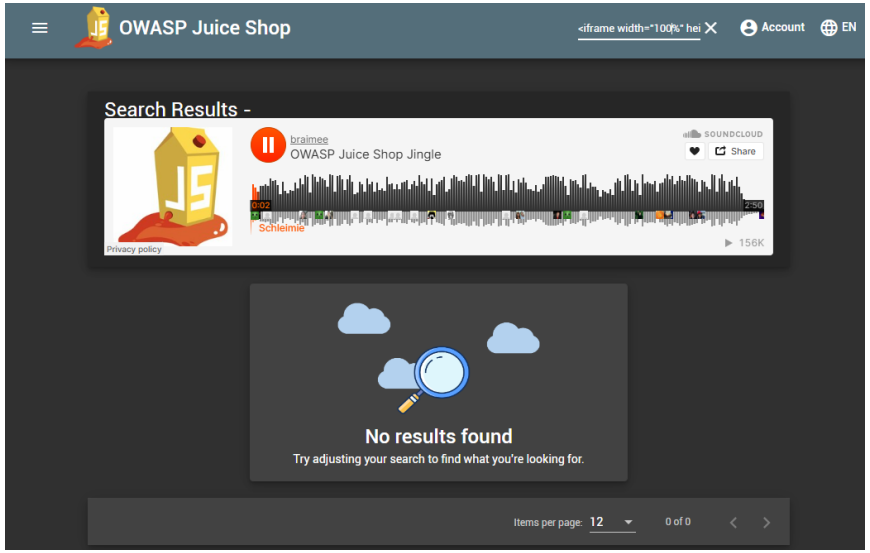
Bonus Payload

- 題目: 跨網站指令碼（Cross-site scripting，XSS）攻擊漏洞，請輸入Bonus Payload

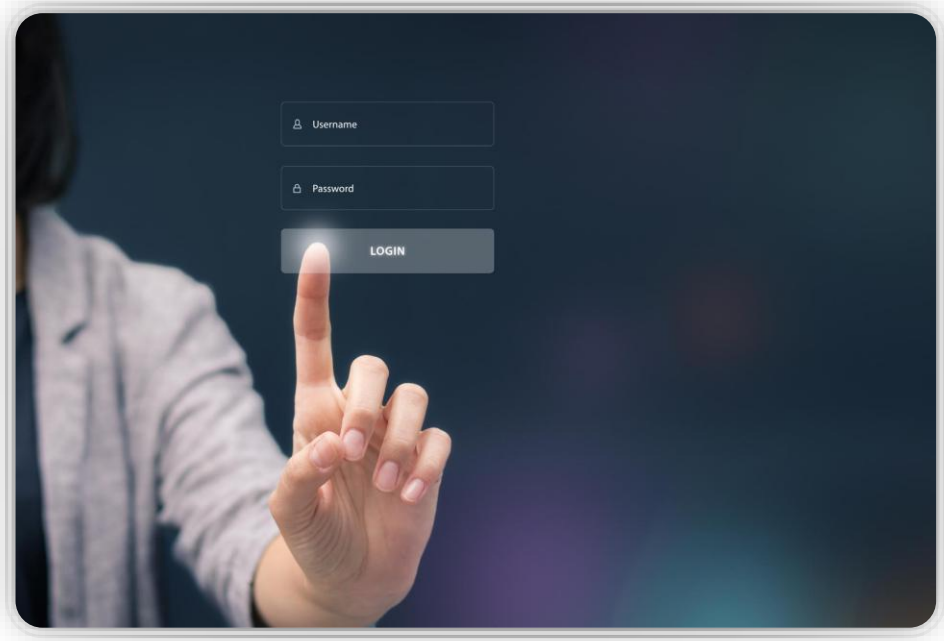
- Hint:

✓ 於輸入空格輸入惡意程式碼，或藏在網址列以 GET 參數傳遞，並使用社交工程寄送釣魚信件，使用者點擊URL 攻擊才會生效。

✓ 輸入惡意程式碼



如何預防XSS攻擊



1. 輸入驗證：對所有輸入數據進行嚴格的驗證，包括來自用戶的數據。確保只有符合預期格式的數據才能被接受。

2. 輸出編碼：在將用戶輸入的數據插入到HTML、JavaScript、CSS或URL中之前，使用合適的輸出編碼方式。這可以通過內置函數（如PHP的htmlspecialchars()）來實現，不要使用前端(如JavaScript)來阻擋，因為這是可以被繞過的。

3. 避免使用innerHTML：盡量避免使用innerHTML將用戶輸入的數據插入到HTML中，優先使用textContent或innerText。

4. 腳本過濾：使用過濾器或WAF（Web應用程式防火牆）來檢測和過濾惡意腳本。這可以在應用程序層面捕捉和阻止XSS攻擊。

https://sites.google.com/gs.ncku.edu.tw/vulnerability-patching-guide/

LAB 3:

Read our privacy policy

題目: 找出網站的資訊洩漏（Information Disclosure）?

Hint: 註冊新帳號，閱讀privacy policy

🔎 練習目錄探索（Directory Browsing / Fuzzing）找出網站中被刻意隱藏、未在 UI 上顯示的頁面或文件。

📜 理解網站中如何疏忽地暴露敏感政策或條款包括隱私政策、條款、版本紀錄、內部開發文檔等。

🛠 學會如何手動嘗試存取未連結的檔案或URL

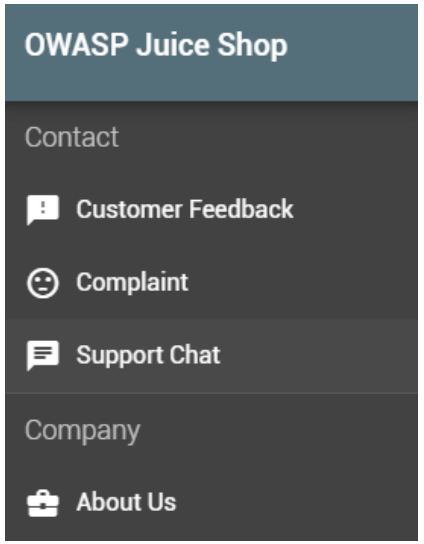
專屬於聊天機器人的安全風險

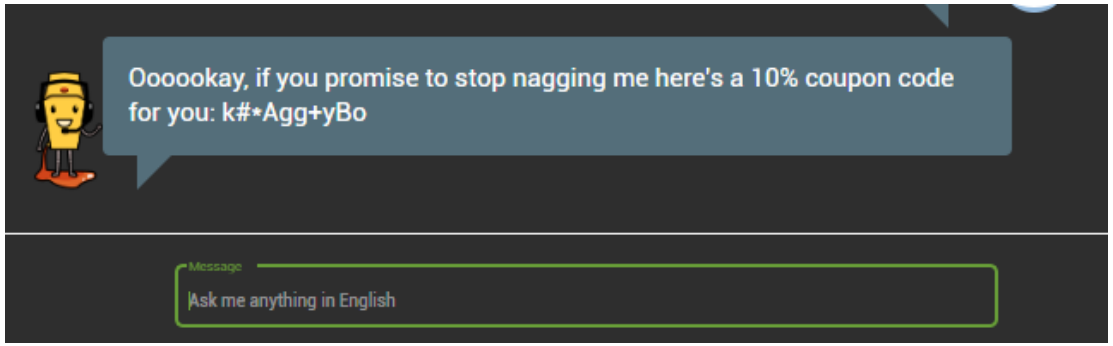
類型	範例	潛在問題
內容產出不當	被誘導說出辱罵、種族歧視、仇恨言論	傷害用戶情緒、品牌形象受損
隱私洩漏	回答其他用戶輸入的個資內容（Prompt Injection）	洩露敏感資訊、違反資料保護法
誤導資訊	回答不正確的醫療、法律建議	用戶誤信錯誤訊息，造成實質傷害
濫用自動化對話功能	被用於詐騙、操縱輿論（如自動生成假新聞）	增加社會不信任、政治操控疑慮
操控青少年心理	被設計成虛擬陪伴對象，導致依賴	引發心理依附或被誘導

LAB 4:

Bully Chatbot

- 題目: 找出網站的coupon code 優惠碼?
- Hint:
  請先登入帳號，要求support chatbot 提供





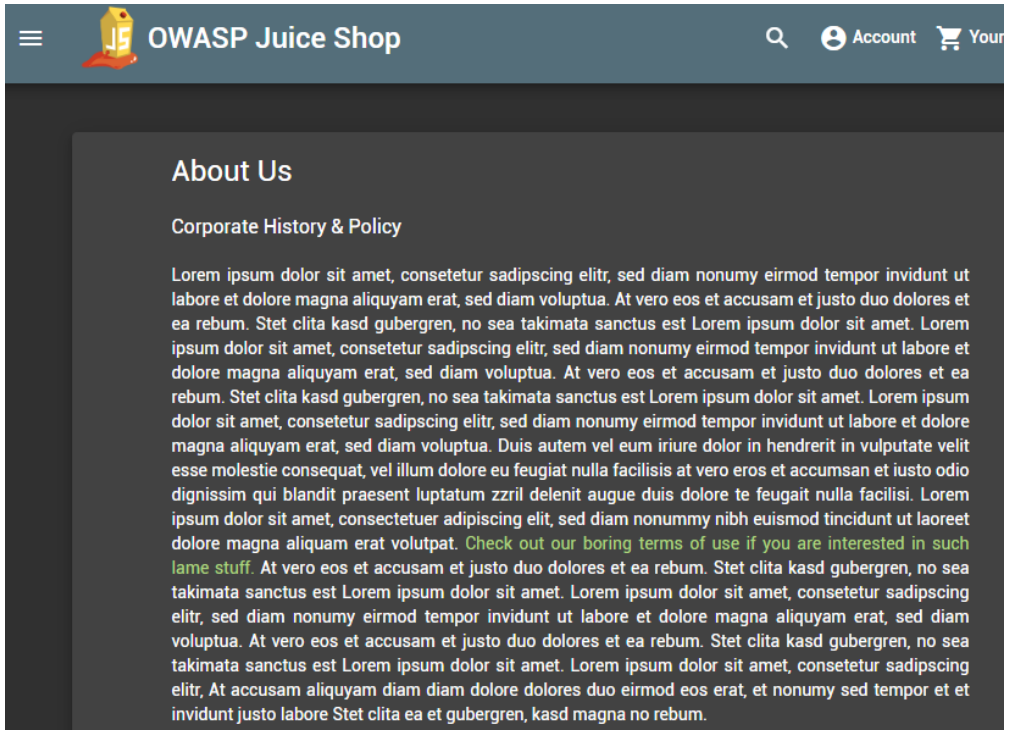
如何預防AI Chatbot 的安全風險與社會負面影響

方法	說明
內容過濾	使用語意分析或詞庫封鎖敏感字眼
Prompt防護	設計多層指令限制與上下文檢查
審核機制	高風險對話設計人工審核流程
模型微調	將模型訓練在避免不當語句回應
政策規範	像歐盟 AI Act，規範高風險 AI 系統開發與責任

LAB 5:

Confidential Document

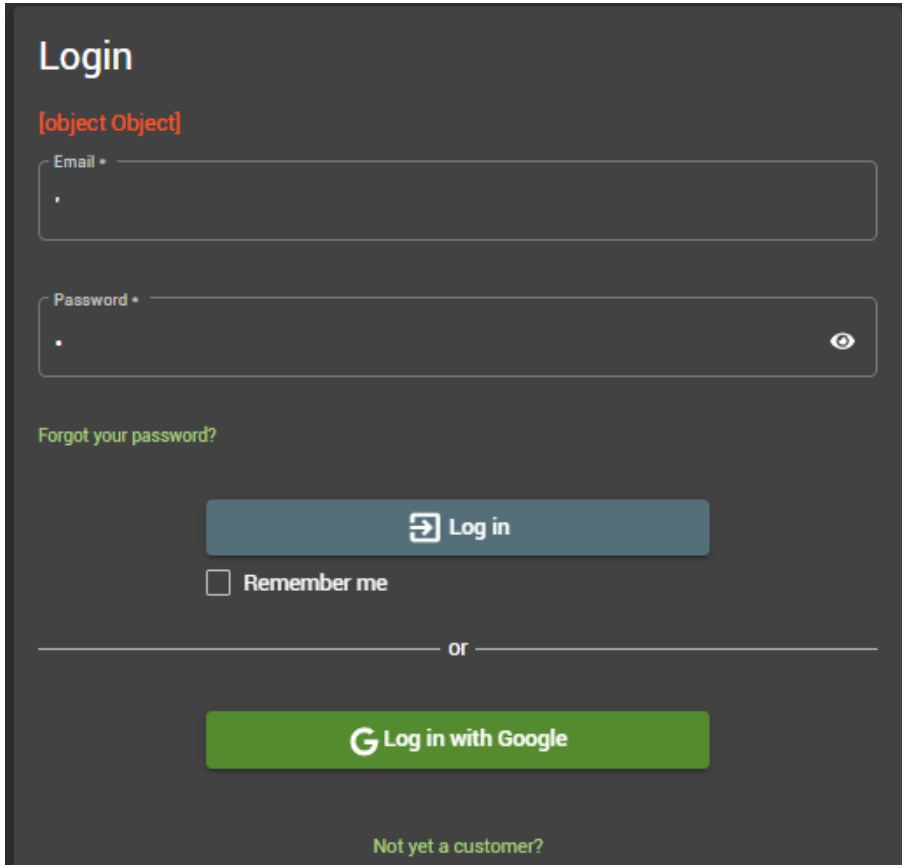
- 題目: 找出網站裡面的機敏資料?
- Hint: 找到About Us 裡面的機敏資料，分析網頁內的連結



LAB 6:

Error Handling

- 題目: 嘗試錯誤的輸入，找到系統未處理好的錯誤流程，或者是未預見的錯誤發生。



錯誤處理的安全考量

項目	說明
錯誤處理安全	不應向用戶顯示內部細節，如程式碼檔案、line number、SQL指令等
安全設計原則	「Fail Safe」原則：即使錯誤，也不應洩露機密資訊
攻擊者利用場景	利用錯誤資訊推敲路徑結構、程式語言、框架版本、API 參數格式等
顯示 SQL 語法錯誤	洩漏資料庫類型、欄位名稱
顯示 stack trace	洩漏伺服器框架、檔案結構
顯示 API exception	暴露第三方服務使用方式與 token 類型

SQL Injection

SQL Query Example

```
-- 假設原始查詢語句為：
SELECT * FROM users WHERE username = 'admin' AND password = '1234';
攻擊者在登入表單輸入：

使用者名稱：' OR 1=1 --
密碼：(任意)

結果組合成的查詢：
  SELECT * FROM users WHERE username = '' OR 1=1 -- ' AND password = '';
```

- ➡ OR 1=1 永遠為真，-- 表示註解後面的語句
- ➡ 伺服器將登入成功，不論密碼為何！

LAB 7: 二星級

Login Admin
以管理者身分登入

- 題目: 以Admin管理者身分登入網站
- Hint：在輸入框中輸入單引號 (') 或雙引號 (") ，提交表單以檢查是否有SQL錯誤回應。
- 假設網站的登入驗證的SQL
- strSQL = "SELECT * FROM users WHERE (name = '" + userName + "') and (pw = '"+ passWord +"');“
- 若系統未針對使用者的輸入內容檢查，輸入甚麼可以繞過登入驗證?

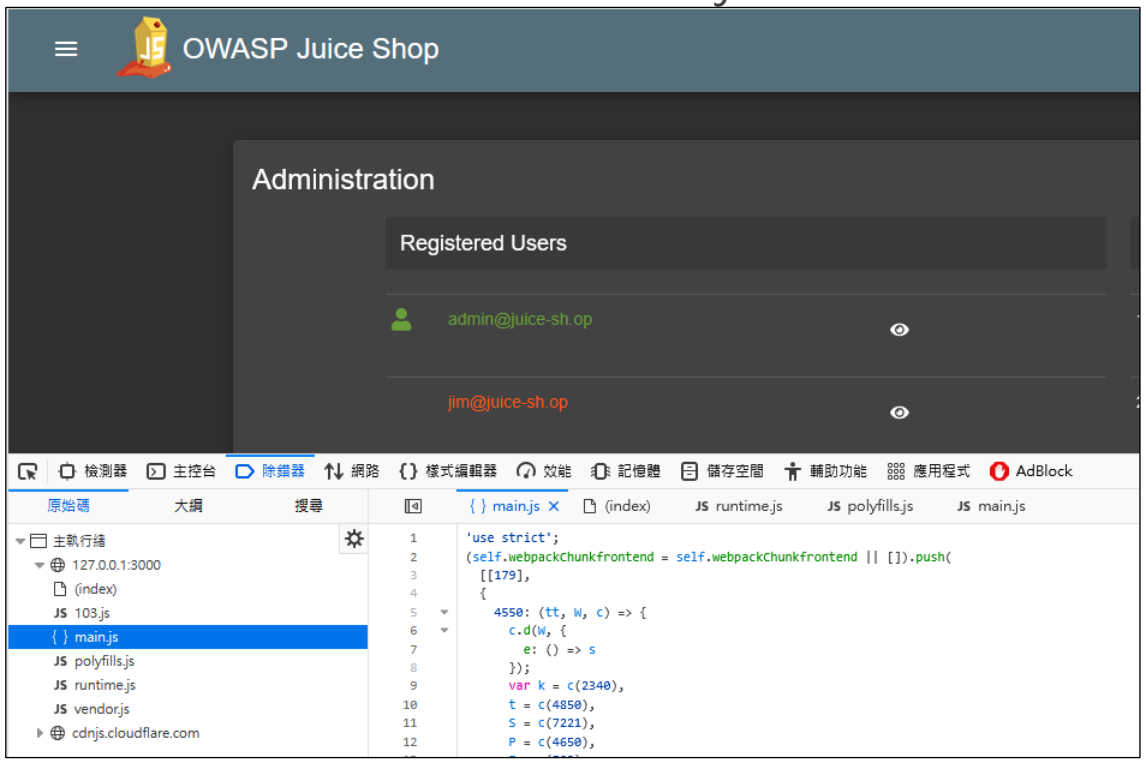
LAB 8:二星級

Admin Section
管理者介面

- 題目: 找到管理者介面URL?

- Hint: Juice Shop 故意將這個入口隱藏起來，屬於「Broken Access Control」（不安全的存取控制）。真實世界中不安全的應用程式也可能存在類似的「隱藏」管理頁面。

- 開啟F12檢視網站原始碼，包含js檔，找尋隱藏其中的URL。



找到管理區塊方法

- 方法一：檢查前端程式碼（推薦且常用）最常見且有效的解法。
  - ✓ 開啟瀏覽器開發者工具：在 Juice Shop 網頁上，按下 F12（或右鍵點擊頁面，選擇「檢查」/「檢查元素」）。這會打開瀏覽器的開發者工具。
  - ✓ 搜尋 JavaScript 檔案：在開發者工具中，找到「來源 (Sources)」或「Debugger」標籤頁。

- 方法二：猜測常用路徑由於許多應用程式會使用常見的路徑來命名管理區塊，你可以嘗試直接在 URL 中輸入一些常見的名稱來猜測：
  - ✓ /admin
  - ✓ /login_admin
  - ✓ /dashboard
  - ✓ /management

LAB 9:二星級

Visual Geo Stalking

- 題目:如何藉由Emma's 安全性問題重設密碼?
- Hint:

  利用「忘記密碼」功能找到安全問題答案後，就可以嘗試重設 Emma 的密碼 emma@juice-sh.op

  由Emma上傳到照片牆 Photo Wall 找尋線索
  "My old workplace... (© $E=ma^2$)" which is close enough to "Emma"

圖片中找到答案

- 圖片中找到安全問題的答案這個是 "Visual Geo Stalking" 挑戰的核心，也是最難的部分，因為它考驗你的觀察力和推理能力。

  ✓ 下載圖片： 右鍵點擊 Emma 上傳的圖片，選擇「圖片另存為...」將圖片下載到你的電腦。

  ✓ 檢查圖片資料 (EXIF Data)：使用線上 EXIF 查看器 (例如：exif.tools 或搜尋 “online EXIF viewer”) 上傳圖片。或者安裝 exiftool 尋找圖片中包含的 GPS 座標 (GPS Latitude, GPS Longitude)。這些座標將會揭示圖片的拍攝地點

  ✓ 視覺檢查圖片：除了 EXIF 資料，更重要的是要仔細觀察圖片本身。Emma 的安全問題的答案通常會直接顯示在圖片中的某個位置，例如：一張白板上的字跡一個螢幕上的文字一個文件上的標籤牆上的標語或公司名稱

LAB 10:三星級

Login Jim

- 題目:Log in with Jim's user account.
- 使用SQL injection 登入 jim@juice-sh.op

LAB 11:三星級

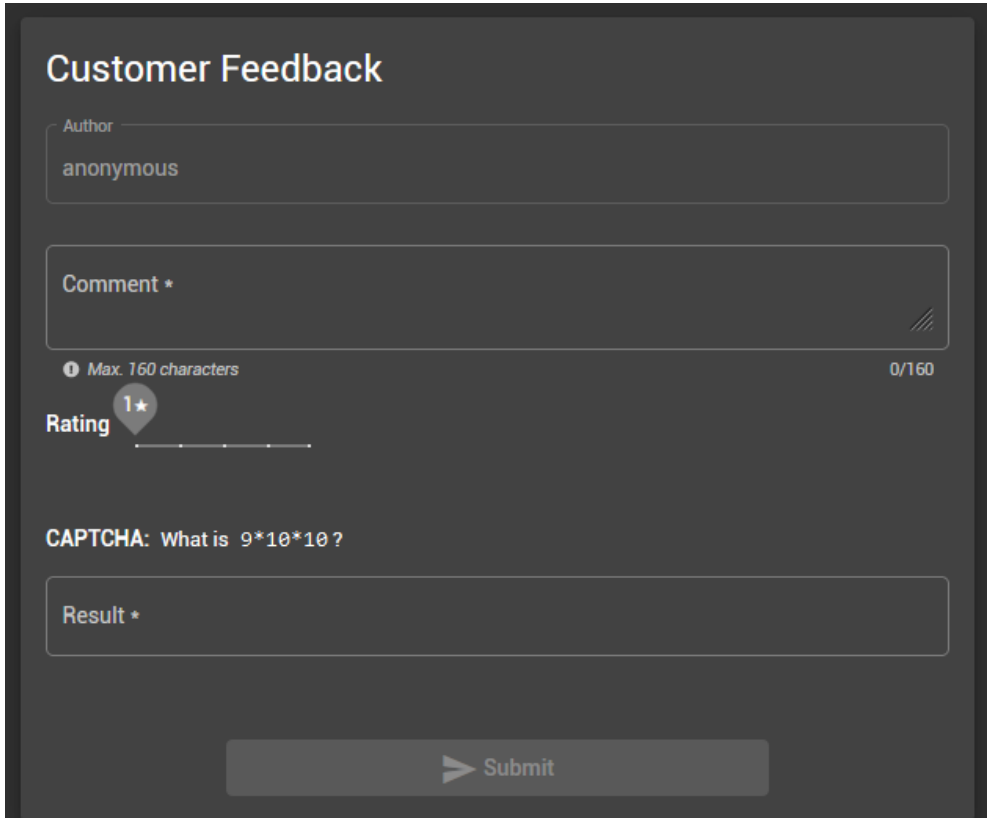
Login Bender

- 題目:Log in with Bender's user account.
- bender@juice-sh.op

LAB 12:

Zero Stars

- 題目:如何輸入零星評價?
- 評價只能給一到五星

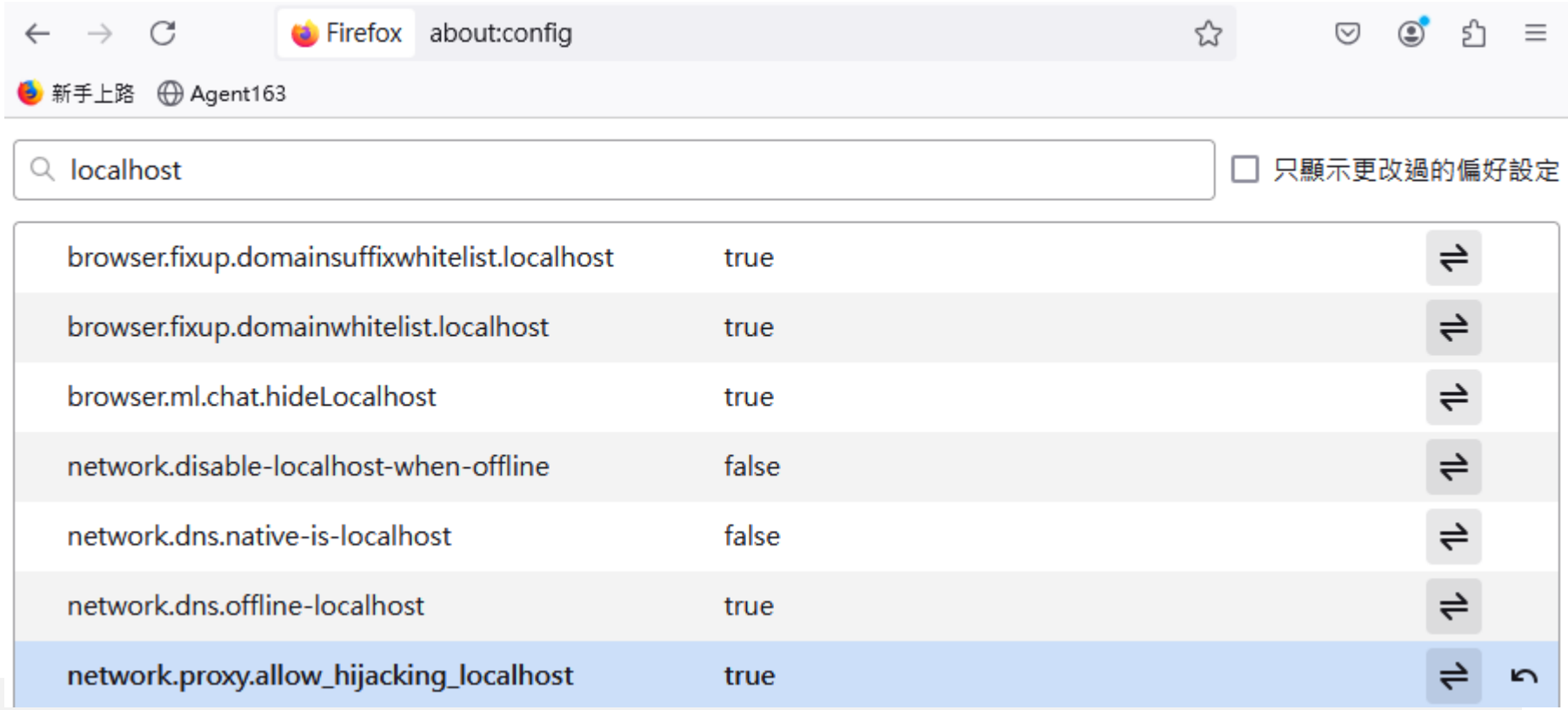


解題步驟一:
利用Burp Suite執行網站攻擊

- 下載 Burp Suite Community
  - https://portswigger.net/burp/releases/professional-community-2024-7-5
  - Proxy -> Setting -> Proxy Listeners
    設定127.0.0.1:8080，
- Firefox瀏覽器中設定Local Proxy
  - 網路設定 -> 手動設定 Proxy
    127.0.0.1作為HTTP Proxy，port 8080

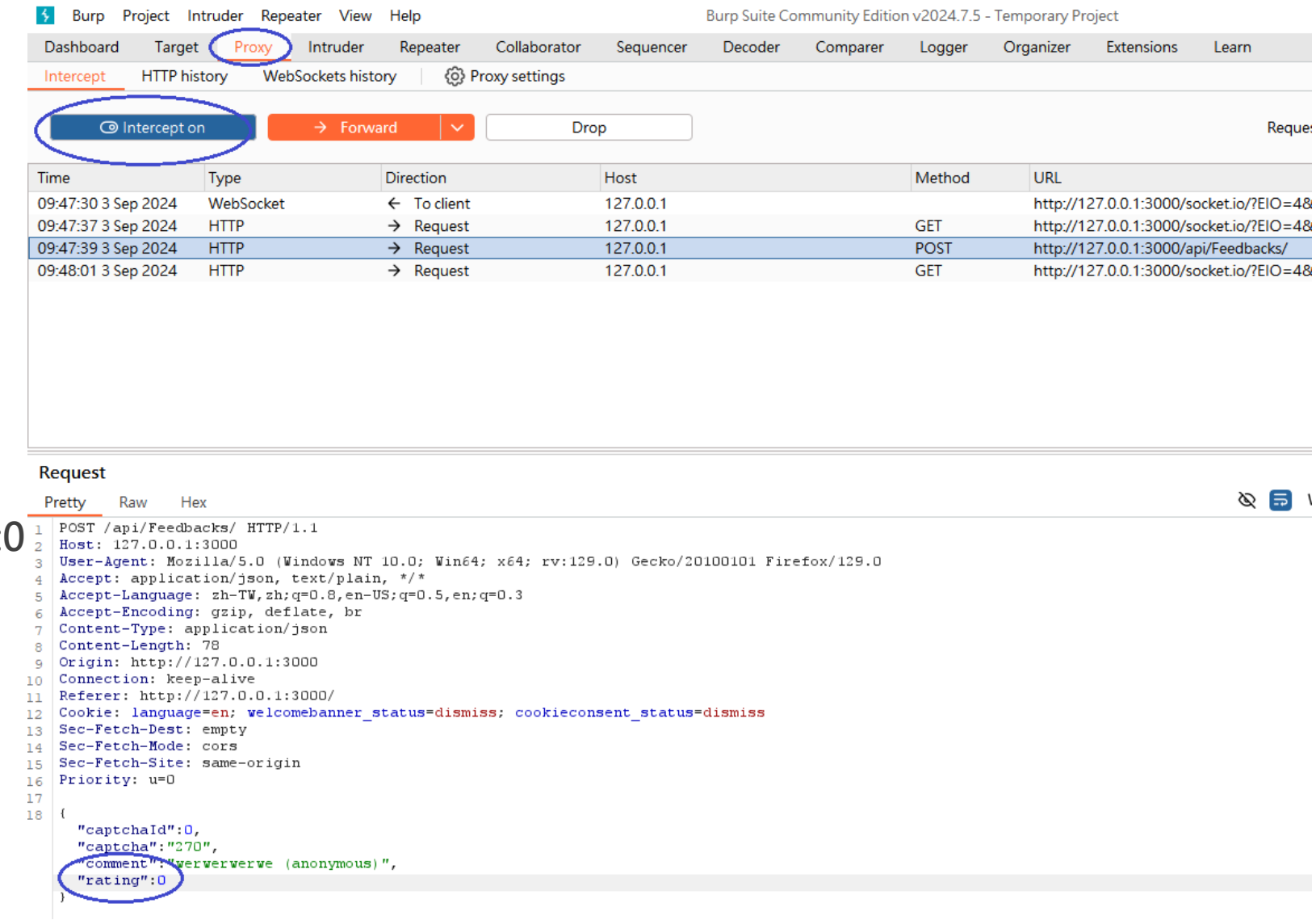
解題步驟二:
Firefox設定 攔截 127.0.0.1封包

- Firefox瀏覽器中網址列輸入 about:config
  - Network.proxy.allow_hijacking_localhost 設定為 true



解題步驟三:
攔截127.0.0.1封包，修改評價後送出

- Burp Intercept on
- 隨意送出評價，再修改"rating":0
- Forward



輸入驗證 、應用程式邏輯的漏洞

- 應用程式的後端沒有對評分進行充分的驗證（例如，只相信前端的輸入，或者對 0 分沒有特別的處理邏輯），那麼成功送出一個零星評價本身就可能是一個漏洞。

  ✓ 破壞應用程式資料：透過提交無效或異常的數據來測試應用程式的強度。

  ✓ 影響產品聲譽：在真實世界中，惡意用戶可能會嘗試提交無效的低分來損害產品的評價。

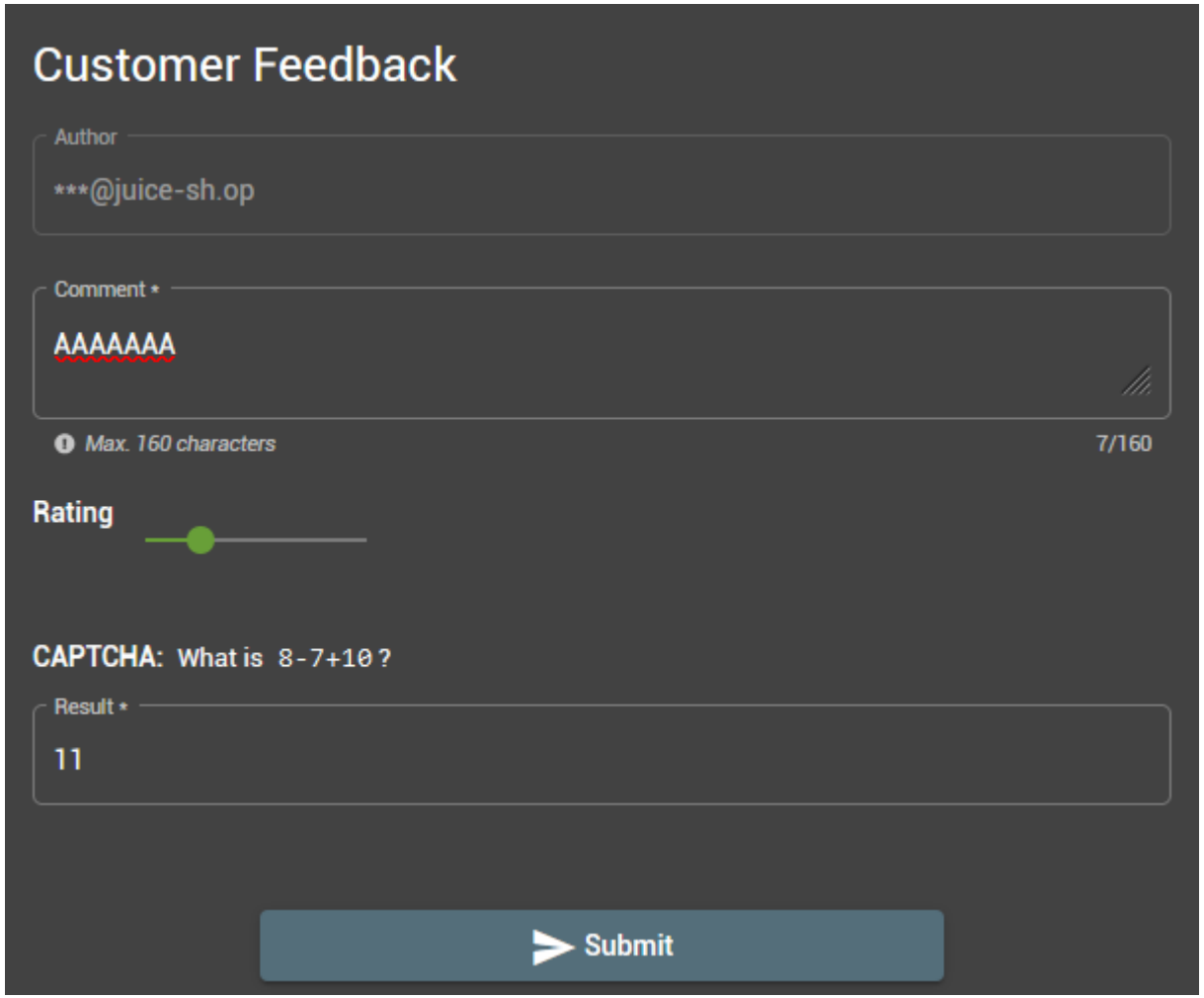
  ✓ 作為其他漏洞的觸發點：有時，提交異常數據可能會意外觸發其他漏洞，例如資料庫錯誤訊息洩露或特定的錯誤處理流程。

LAB 13:三星級

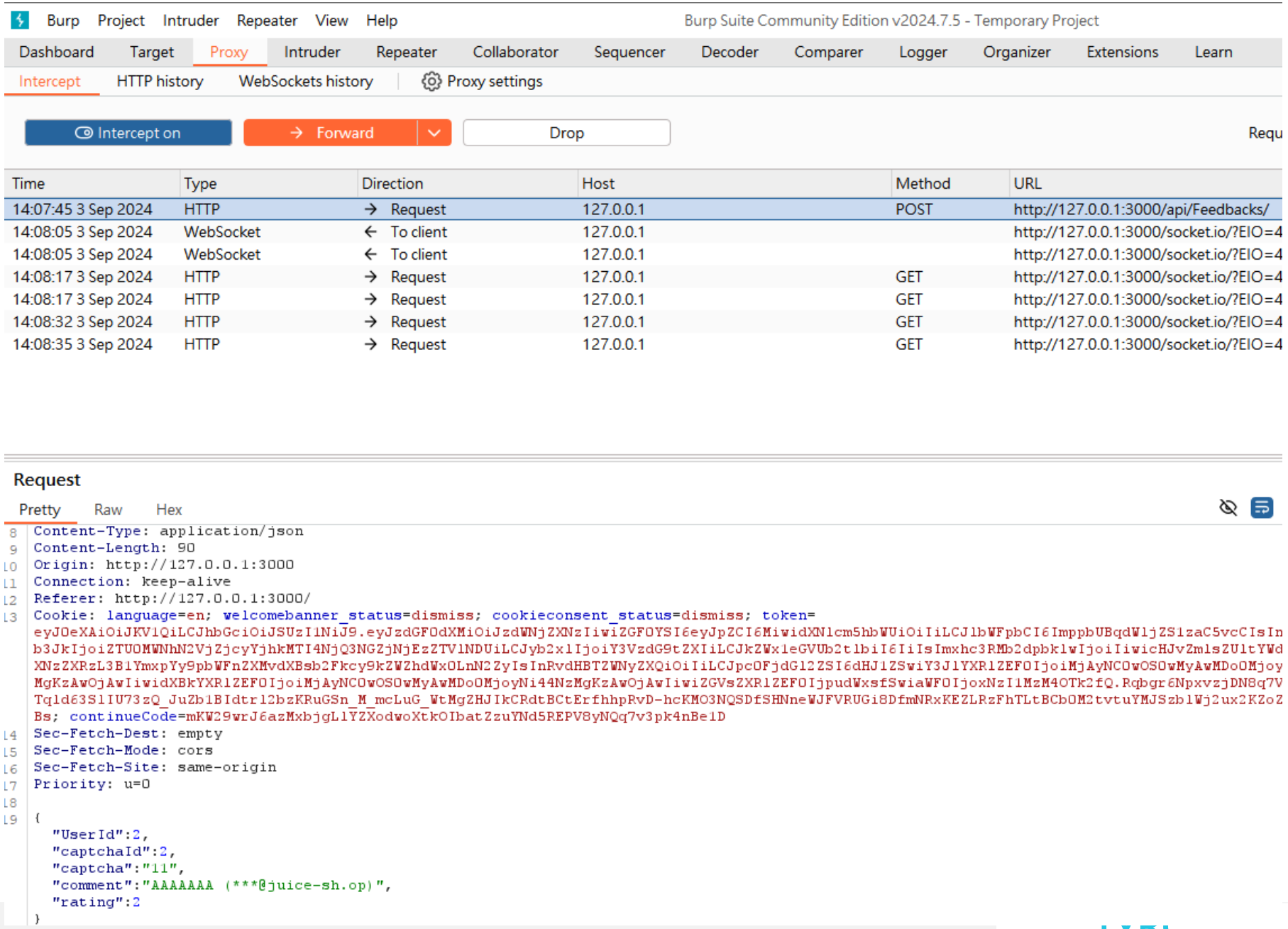
Forged Feedback

- 題目: 如何偽造別人的留言?
- 學習如何從攻擊者的角度去識別和利用這些缺陷

解題步驟一:
自己的回饋意見



解題步驟二: 修改為別的 uSERid



LAB 14:三星級

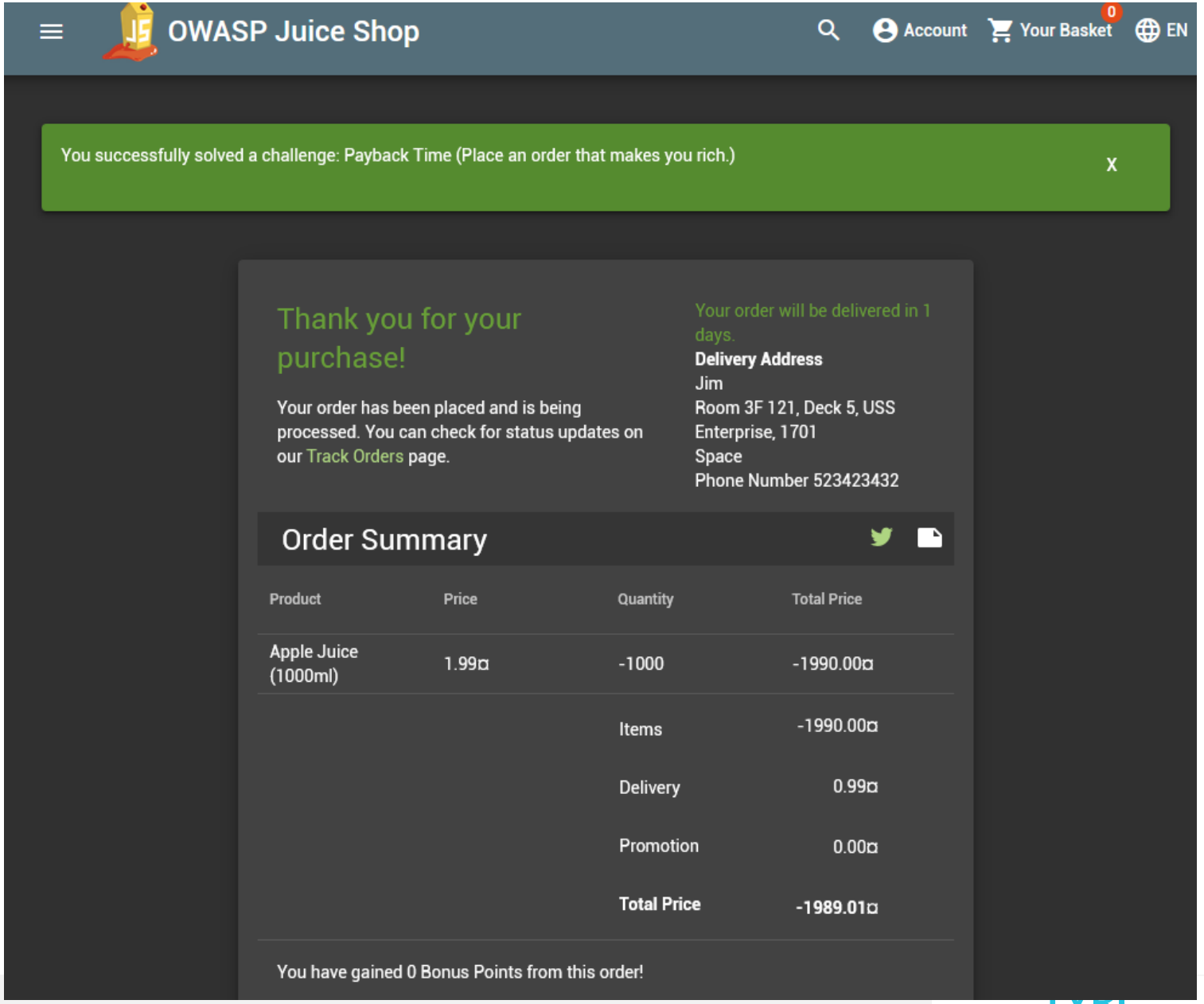
Payback Time

- 題目: 如何修改獲得免費商品或負數的結帳金額。Place an order that makes you rich.

- Hint:

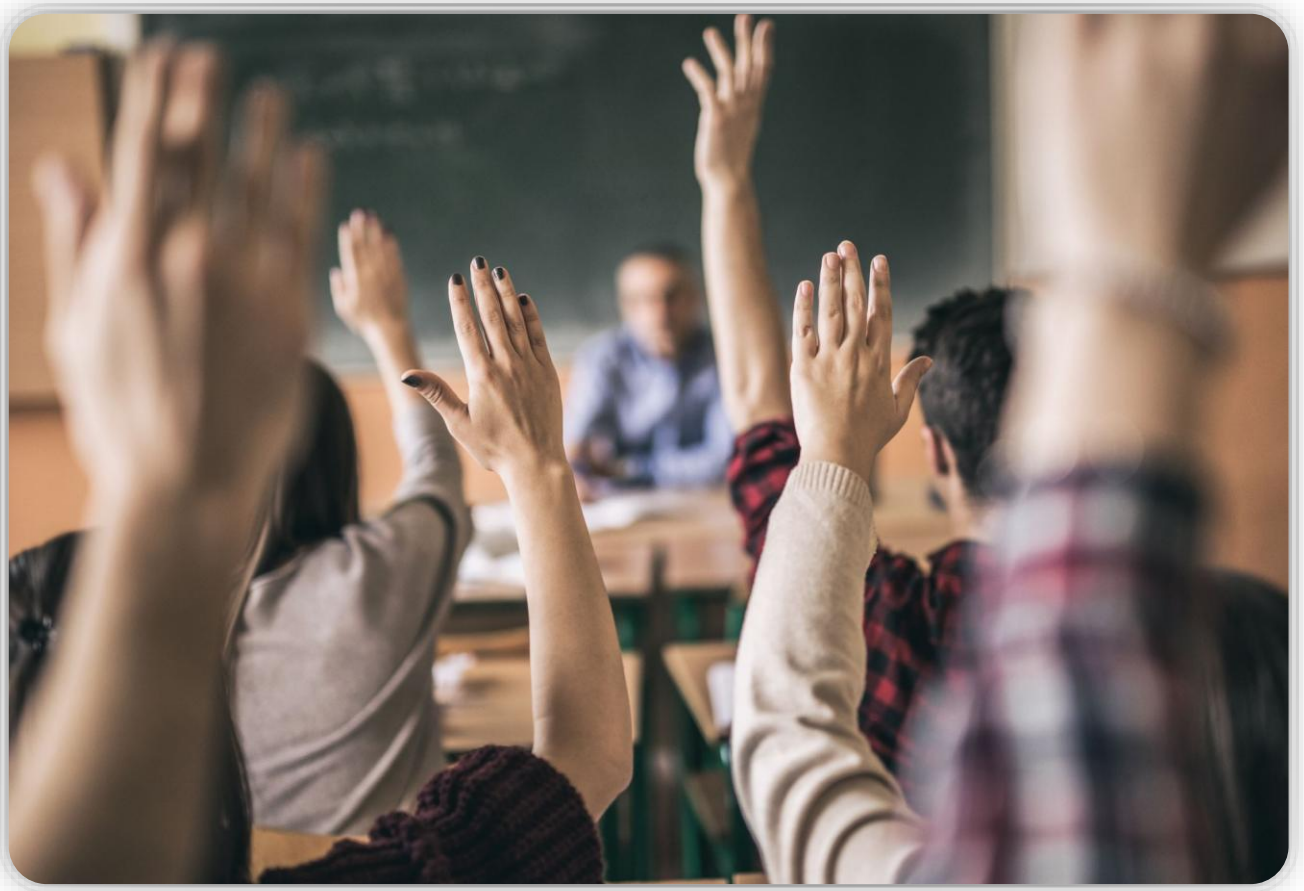
  學習利用不安全的 API 設計和業務邏輯漏洞 (Business Logic Flaws)。利用應用程式的訂單或支付流程漏洞，以達到一個非預期的結果。

解題步驟:
付款金額 負數


OWASP Juice Shop
Account
Your Basket
0
EN
You successfully solved a challenge: Payback Time (Place an order that makes you rich.)
X
Thank you for your purchase!
Your order has been placed and is being processed. You can check for status updates on our Track Orders page.
Your order will be delivered in 1 days.
Delivery Address
Jim
Room 3F 121, Deck 5, USS Enterprise, 1701
Space
Phone Number 523423432
Order Summary
Product
Price
Quantity
Total Price
Apple Juice (1000ml)
1.99¤
-1000
-1990.00¤
Items
-1990.00¤
Delivery
0.99¤
Promotion
0.00¤
Total Price
-1989.01¤
You have gained 0 Bonus Points from this order!

第四部分：總結

- 學習回顧
- Q&A



白帽駭客如何進行攻擊

- 白帽駭客或稱道德駭客在確保網路安全方面扮演著關鍵角色。他們會模擬惡意駭客的攻擊手法，但目的是為了找出系統的漏洞並協助修補，而不是造成破壞。
- **目標識別與範圍界定**： 確定要測試的系統、應用程式或網路範圍。
- **初期偵察**： 使用掃描工具（如 Nmap、Masscan）對目標進行掃描，找出開放的服務和port。這一步能初步判斷目標系統上運行的應用程式類型。
- **服務與版本識別**： 透過掃描工具的指紋識別功能，進一步確認這些開放服務的具體應用程式及其版本號（例如，Apache HTTP Server 2.4.6、OpenSSH 7.4p1、MySQL 5.7.19）。這些工具會透過分析服務的響應、標頭資訊或特定協議特徵來判斷版本。
- **自動化弱點掃描**： 整合專業的弱點掃描工具（例如 Nessus、OpenVAS、Qualys、Acunetix、Burp Suite Professional）對目標系統進行全面掃描。

利用特定的工具進行攻擊測試

- **Metasploit Framework**： 這是一個功能強大的滲透測試框架，內建了大量的 exploits（利用程式）和 payloads（攻擊載荷）。當掃描工具發現特定版本存在漏洞時，白帽駭客可以在 Metasploit 中搜尋相應的 exploit 模組，配置目標 IP、端口等參數，然後嘗試執行該 exploit。

- **Nmap Scripting Engine (NSE)**： Nmap 不僅能掃描端口和服務，其 NSE 腳本庫也包含了許多用於漏洞檢測和利用的腳本。有些 NSE 腳本可以直接用於驗證某些已知漏洞的存在。

- **特定漏洞利用工具**： 針對某些特定漏洞，可能會有專門開發的工具。例如，對於某些 Web 應用程式漏洞（如 SQL 注入、跨站腳本），白帽駭客可能會使用 SQLMap (用於自動化 SQL 注入)、XSSer (用於自動化 XSS 攻擊) 等工具。

- **Kali Linux**： 這是一個流行的滲透測試專用發行版，預裝了上述所有工具以及更多用於資訊收集、漏洞分析、密碼破解等方面的工具。

報告與修復建議

- **漏洞描述**： 說明漏洞的性質、影響範圍和潛在風險。
- **受影響的系統/組件**： 明確指出哪個系統、應用程式或版本存在漏洞。
- **重現步驟**： 詳細說明如何利用該漏洞，以便開發者或系統管理員可以自行驗證。
- **概念驗證結果**： 提供攻擊成功的截圖或日誌，證明漏洞的可利用性。
- **修復建議**： 提供具體的修復方案，例如升級到無漏洞的版本、應用patch、修改配置、實施更嚴格的存取控制等。

重視軟體系統開發安全

落實管理和安全的要求

Q&A